REMARKS

Claims 1-17 now stand in the application, new claims 12-17 having been added.

In the Request For Rehearing filed , applicants pointed out to the Board that the word "contain" can mean contain in the sense that it incorporates something as part of, or it can mean to contain in the sense of encapsulating, but when used in conjunction with the word "container" the meaning is constrained to the second meaning, i.e., encapsulation. In its Decision, the Board quoted the dictionary definition of "contain", showing either of the two usages. The Board did not consider how the contextual use with e term "container" would limit which of the two normal usages makes sense. The issues is now obviated by amending the independent claims to clarify that the containers encapsulate the service component. This is supported in the specification at line 5 from the bottom of page 4 (in the later part of the paragraph numbered [0010] in the publication US 2002/0003868 corresponding to the present application) and at the first line of page 16 (the first line of the paragraph numbered [0040] in US 2002/0003868).

In rejecting claim 1 in the earlier Office actions, the examiner considered the Terminal Agent 102 of Yates et al to correspond to the claimed service computer. A claimed function of the service computer is to execute a service machine. Yates describes the terminal agent as executing software modules, which the examiner equates with the claimed service containers.

Columns 17-18 of Yates explain that each agent is made up of plural component objects, which cooperate to provide the service system. The operation of each object is governed by policies that may be embedded within the object itself or may instead be loaded by the object during operation. (lines 32-47 of column 17) There are three different types of objects: service independent building blocks (SIBBs), adaptors and co-ordinators. (lines 13-21 of column 17) A

SIBB cannot by itself provide a service. Otherwise, it would not be labeled as a ***service independent*** building block. As explained at column 18, it is a combination of SIBBs and adaptors, coordinated by a coordinator, and all operating in accordance with policies, that provides a service to the user.

The examiner equates the "code and SIBBs" of a "module" in Yates et al with the claimed service machine, while at the same time equating the module itself with the claimed "container." But it is clear that in the description of Yates, a SIBB ***is*** a module (see, e.g., lines 3-5 of column 4), it is not something contained within a module.

Returning to the language of claim 1, it requires that service container containing a service machine is transmitted from a service server to a service computer. The issue of where the container comes from will be discussed later, but for the moment it is enough to note that the terminal agent in Yates must receive a container containing a service machine. The agent in Yates does not receive such a container. At best, it receives individual pieces of software that it builds into a machine using a coordinator and an appropriate combination of atomic or compound SIBBs, adaptors and policies.

It may be that the service component referred to in the last subparagraph of claim 1 can be read on a SIBB, or more likely, a compound SIBB. But neither the service machine nor that service component are described anywhere in Yates et al as being sent to the terminal agent in a container.

The examiner relies on Beck for this teaching, but it is absent there as well. Beck teaches downloading of service code. There is no discussion of a container that contains a service machine. Nor is there any discussion of a service component being sent in a container, which

component will either be executed by or applied by the service machine sent in the first container. If the teachings of Beck were considered by one of skill in the art, it would at best have suggested (as has been alleged by the examiner) that the modules needed by the terminal agent in Yates et al could be obtained by downloading instead of being stored locally. This would be consistent with Yates et al describing that the agents may have access to a shared set of software elements. But the result of this combination/modification would still be that the agent in Yates et al would download a coordinator and whatever SIBBs and adaptors and policies are needed to configure itself top provide the needed service. There would be no transmission of a container containing a service machine and optionally a service component to be applied or executed by the service machine.

Yates et al describes reconfiguring agents to provide services as needed. The present invention is directed to something different, trying to eliminate the dependency of provided services on the architecture or capability of the network to which the user is presently connected. So there is on one level a general similarity of concept, i.e., software-configured service provision, but the manner in which the concept is implemented/applied differs because the purposes are quite different.

The present invention is not looking to maximize the ability to reconfigure a service provision by only having to change one module, but runs directly contrary to the teaching to Yates et al by putting a service machine into a container and sending it to the service computer. For Yates to use containers in the manner claimed would run contrary to a stated goal of Yates.

Further, the present invention contemplates that the user may be connected to the service computer through any of plural different networks requiring different interfaces, so it is

necessary for the service computer to provide a network lock specific to the particular network being used and it is necessary for the service container to be adapted to use this network lock. This is not an issue in Yates et al. The distributed processing environment (DPE) in Yates is described at line 8 of column 15 as being used for handling communication ***between agents***. There is no discussion anywhere of the agent receiving a service container that is adapted to use a predefined interface of a network lock provided by the service computer.

For the above reasons, it is submitted that the invention defined in claim 1 would not have resulted from any obvious combination of the teachings of Yates et al and Beck.

The remaining independent claims distinguish over the cited art for the same reasons.

New claims 12-17 are added to emphasize the significance of the network lock provision.

In view of the above, reconsideration and allowance of this application are now believed to be in order, and such actions are hereby solicited. If any points remain in issue which the Examiner feels may be best resolved through a personal or telephone interview, the Examiner is kindly requested to contact the undersigned at the telephone number listed below.

Respectfully submitted,

/DJCushing/

SUGHRUE MION, PLLC
Telephone: (202) 293-7060
Facsimile: (202) 293-7860

WASHINGTON OFFICE
23373
CUSTOMER NUMBER

David J. Cushing
Registration No. 28,703

Date: November 6, 2007